ENHANCING SPACECRAFT AUTONOMOUS OPERATIONS VIA MACHINE LEARNING

GABRIELE MARTINO INFANTOLINO

POLITECNICO
MILANO 1863

APPLICATION OF SUPPORT VECTOR MACHINES TO SOLAR GENERATOR FAULT DETECTION AND SPACE TRAFFIC MANAGEMENT

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science
in
Space Engineering

April 2018

Gabriele Martino Infantolino: *Enhancing Spacecraft Autonomous Operations via Machine Learning,* Application of Support Vector Machines to Solar Generator Fault Detection and Space Traffic Management, Master of Science, © April 2018

SUPERVISOR:
Franco Bernelli Zazzera

CO-SUPERVISOR:
Pierluigi Di Lizia

LOCATION:
Politecnico di Milano

TIME FRAME:
April 2018

Ohana means family.
Family means nobody gets left behind, or forgotten.

— Lilo & Stitch

Dedicated to the loving memory of my grandfather Giuseppe.
1939 – 2005

ABSTRACT

The definition of on-board autonomy as from the European ECSS Space Segment Operability Standard recites: *On-board autonomy management addresses all aspects of on-board autonomous functions that provide the space segment with the capability to continue mission operations and to survive critical situations without relying on ground segment intervention*. Environment with high uncertainty, limited on-board resources, limited communications, criticality are all factors that influence the level of autonomy. On the other hand, modern days have seen the rise in popularity of Machine Learning algorithms, but their decision-making effectiveness is not yet extensively studied in the space domain. The present thesis explores the applicability of Machine Learning models, specifically Support Vector Machines (SVM), in two different cases. Failure detection and identification is an issue that must be efficiently tackled during the operational lifetime of spacecraft. Many of them provide an abundance of system status telemetry that is monitored in real time by ground personnel and archived to allow for further analysis. Recent developments in data mining and machine learning for anomaly detection make it possible to use the wealth of archived system data to produce sophisticated system health monitoring applications, that can run autonomously on-board spacecraft. Archived data, as well as data produced with dedicated numerical simulations, are used to train intelligent systems to automatically detect anomalous time series of the produced telemetry, recognize their possible correlation to a system failure, and classify the failure. The test case is the monitoring of Rosetta's lander Philae solar power generator. A complete model of both the cometary environment and the solar panels has been developed, in order to simulate the real telemetry. The training data, generated for nominal and faulty cases, are then used to train an SVM-based classifier, with the goal of classifying permanent (broken solar cells) and temporary (partial shading) power loss conditions. The telemetry obtained during simulated cometary days, either entirely nominal or including anomalies, is then fed to the classifier to test its performance and to identify the minimum number of measurements that is necessary for a successful classification of the failures of interest. The second application revolves around joining the broad topic of Space Traffic Management (STM) with Machine Learning. Starting from a literature review of the needs and criticalities in the STM domain, and taking inspiration from existing applications in the Aeronautic sector, an approach to spacecraft collision warning is proposed and simulated.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Part I

THESIS BACKGROUND

1 THE ROLE OF AUTONOMY FOR SPACE EXPLORATION

1.1 INTRODUCTION

A testament to the ancestral human desire to explore and understand the surrounding world, space exploration is leading humans great distances away from Earth but ultimately closer to its true understanding, with an ever-increasing number of artificial satellites and probes. A realization emerges, that smarter systems must be developed that can respond to the uncertain environment in which they operate, with limited or non-existent human intervention. In addition, there is the compelling urge to reduce the overall cost of operating in space, and automating vehicles operations and maintenance is a key factor (Frost [15], Jónsoon et al. [22]).

A fundamental difference exists between *autonomous* and *automated* systems, as far as decisional capability is concerned, as remarked in [15]: an automated system doesn't make choices for itself but follows a (potentially sophisticated) script. If the system encounters an unplanned-for situation, it stops and waits for human input. Thus for an automated system, choices have either already been made and encoded, or they must be made externally. By contrast, an autonomous system does make choices on its own, even when encountering uncertainty or unanticipated events. An *intelligent* autonomous system makes choices using sophisticated mechanisms that are judged by the quality of the choices it makes.

The notion of autonomy in the space sector has generally been restricted to predefined explicit behaviors and programs, i.e. automation. However, it breaks down under increasing uncertainty and nondeterminism, such as when navigating on a planetary surface or investigating unpredictable and transient events. Higher levels of autonomy and automation using Machine Learning technologies enable a wider variety of more capable missions and enable humans to focus on different tasks. Among the others (Jónsoon et al. [22]), two potential applications that can greatly benefit from such technologies are hereby presented, followed by an introductory overview of Machine Learning.

1.2 FAULT DETECTION AND HEALTH MANAGEMENT

The health management decision-making capability addresses as a whole the need for detecting, diagnosing and reacting to events occurring inside a system, given streams of observations, through the use of behavioral models. Currently, health status information for spacecraft is often obtained through limit thresholds and simple algorithms that reason locally about an individual component. While this - the need for automation - may be not a real issue for Earth-orbiting spacecraft, which promptly communicate with ground stations, it becomes more and more important as the focus is moved on interplanetary space missions. A perfect example is the well-known Rosetta mission (see Section 3.2). It was not possible to intervene in real-time during operations, due to the communication time lag. An higher degree of autonomy would have been favorable, especially for the lander Philae, that operated in an unknown and harsh environment. Considering the task of fault *detection*, an AI-based model will be able to operate within the real-time execution loop, thus permitting the system (as instructed by the follow-up health management chain) to respond quickly based on a supposedly true identification of the source of the anomaly. Consequently, the diagnostic system must be smart enough to also distinguish between changes caused by faulty components from transient state changes caused by the environment.

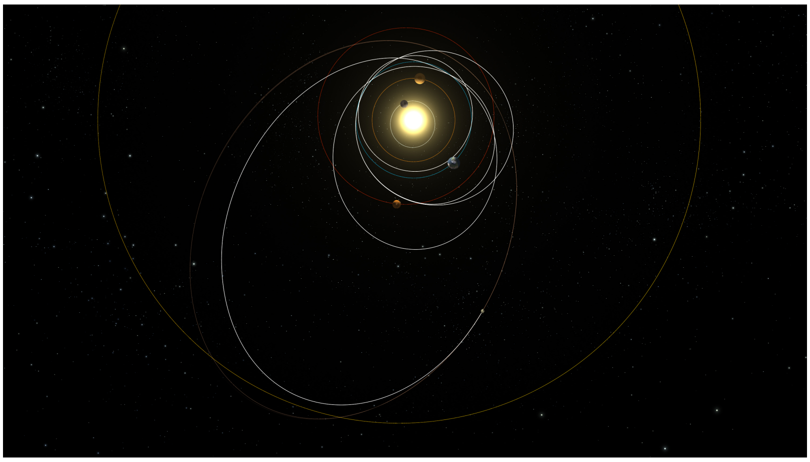


Figure 1: Rosetta and Philae's ten year long journey. ©ESA-C. Carreau

1.3 SPACE TRAFFIC MANAGEMENT

The phrases *Orbital Traffic Management* and *Space Traffic Management* (STM) have been used interchangeably in the literature, public discourse, and policy discussion to describe governance approaches for supporting the safety of orbiting spacecraft and debris mitigation [7]. With an exponentially growing number of space-related practical services and research interests, a new focus has been appropriately made on the defense and protection of spacecraft to ensure the continued flow of information (Cukurtepe and Akgun [13], Jah [20], Brown, Cotton, et al. [7], Contant-Jorgenson, Lála, Schrogl, et al. [11]). A few definitions have been proposed but the process is still on-going. It appears that analyzing the applicability of AI technologies to something as vague as STM is not a straightforward process.

Let us clarify at first the possible course of action and area of interest. Space Traffic, as defined in 2001, *encompasses all the phases of a space object's life, from launch to disposal. It consists of activities intended to prevent damage in the near term (such as Collision Avoidance and coordination of reentry) as well as actions that must be taken to reduce the long term potential for future damage (such as de-orbiting or moving satellites into disposal orbits)* [6]. As stated in [7], the term space traffic management should include both policy *and* the tools and processes used for Space Situational Awareness (SSA). It is assumed the two cannot be separated in the context of the goal of space traffic safety. Several critical areas for research, such as (but not limited to) space environment effects and impacts on space objects and events, Big Data storage, management and exploitation, Behavioral Science, are listed in [20].

Also, *space traffic management* creates a direct analog to *air traffic management*. That is one possible framework to approach orbital safety issues [7]. In particular, considering all the due differences of the case, one can think of extrapolating what it is done in the aeronautic research field, regarding traffic control, and make a parallel in the space branch. An interesting study is the one from Kochenderfer and Chryssanthacopoulos [24] and Julian, Lopez, et al. [23]. To design the decision making logic for aircraft collision avoidance systems for both manned and unmanned aircraft, recent work formulates the problem as a partially observable Markov decision process (POMDP). That has led to the development of the ACAS X family of collision avoidance systems. The variant for unmanned aircraft, ACAS Xu, uses dynamic programming to determine horizontal or vertical resolution advisories in order to avoid collisions while minimizing disruptive alerts [24].

The dynamic programming process for creating the ACAS Xu horizontal decision making logic results in a large numeric lookup table that contains scores associated with different maneuvers from millions of different discrete states. Due to on-board hardware limitations (same applies for spacecraft), a deep neural network, trained using supervised learning (see Section 1.4), is adopted to learn a complex non-linear function approximation of the table, to improve storage efficiency. Part of the present work is then dedicated to a preliminary study and assessment of the feasibility of transferring a similar concept to the space domain.

1.4 OVERVIEW OF MACHINE LEARNING

Wearing now the hat of the computer scientist, present days can be definitely called the era of big data. Every digital process and social media exchange produces it. Systems, sensors and mobile devices transmit it. Big data arrives from multiple sources and it calls for automated methods of data analysis, which is what Machine Learning offers (Murphy [27]).

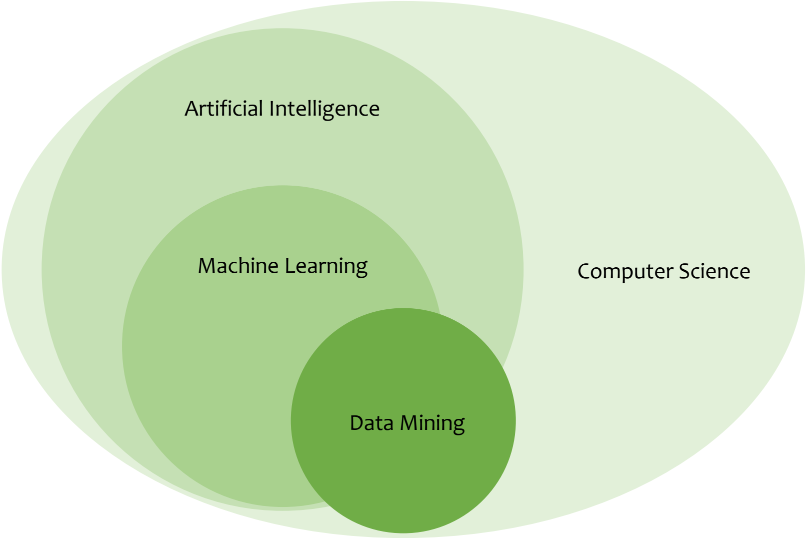


Figure 2: How Machine Learning fits into Computer Science

Figure 2 depicts only part of the ramifications within CS/AI fields

A widely quoted, operational definition of machine learning, introduced by Tom M. Mitchell[1], is the following: *"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E"*. Generally speaking, machine learning provides a set of methods that can automatically detect patterns in data, and then use the uncovered patterns to predict future data, or to perform other kinds of decision making under uncertainty. To better understand what machine learning can do in practice, while leaning towards the description of the application in the present work, it is useful to classify Machine Learning tasks into three broad categories. The classification depends on the learning methodology:

SUPERVISED LEARNING The computer program is presented with a known set of inputs and related outputs, and the goal is to learn a general rule that maps unknown inputs to the correct outputs.

UNSUPERVISED LEARNING The learning algorithm is on its own to find structure in its input, which can be a goal in itself or an intermediate step.

[1] Tom Michael Mitchell (born August 9, 1951) is an American computer scientist and E. Fredkin University Professor at the Carnegie Mellon University (CMU). He is well known for his contributions to the advancement of machine learning, artificial intelligence, and cognitive neuroscience

REINFORCEMENT LEARNING The computer program interacts with a dynamic environment and must reach a certain goal (such as driving a vehicle or playing a game against an opponent). The program is provided feedback in terms of rewards and punishments as it navigates its problem space.

The focus in this work is on Support Vector Machines (SVM), which are *supervised* learning models with associated learning algorithms, used for classification and regression analysis. Without dwelling on theoretical details (see Chapter 2, Murphy [27], Cortes and Vapnik [12]), at first it is enough to point out that SVM are *sparse kernelized* machines: the solution depends only on a subset of the training data and the variables are defined in terms of a kernel function.

1.5 THESIS OUTLINE

The thesis is organized as follows. Chapter 2 gives an introduction to Support Vector Machines for binary and multi-class classification. The procedure of training and validating a Machine Learning model is also explained. Chapter 3 presents a brief overview of Rosetta mission and Philae main characteristics. The modeling assumptions and methodology are described, showing how the simulated telemetry is produced. Chapter 4 reports the analysis of the results of the first part. Chapter 5 gives an introduction to the broad issue of Space Traffic Management, addressing needs and criticalities and explaining the relationship with spacecraft collision avoidance. State-of-the-art procedure are briefly explained and the proposed approach for collision warning is then described. Chapter 6 reports the analysis of the results of the second part. Chapter 7 concludes the thesis and discusses some areas for future work.

2 SUPPORT VECTOR MACHINES

This chapter is intended to provide introduction to Support Vector Machines and addresses needs and criticalities of building a Machine Learning model as a whole.

2.1 INTRODUCTION

Developed in the '90s by Vladimir Vapnik and his team at AT&T Bell Labs. (Cortes and Vapnik [12]), SVM are widely adopted in machine learning, statistics, and signal processing. They have been applied to solve a variety of practical problems in different fields of science, such as robotics, computer vision, pattern recognition, computer security, neural and medical image analysis, soft biometrics, etc. They are based on the concept of decision planes that define decision boundaries. A decision plane is one that separates between a set of objects having different class memberships.

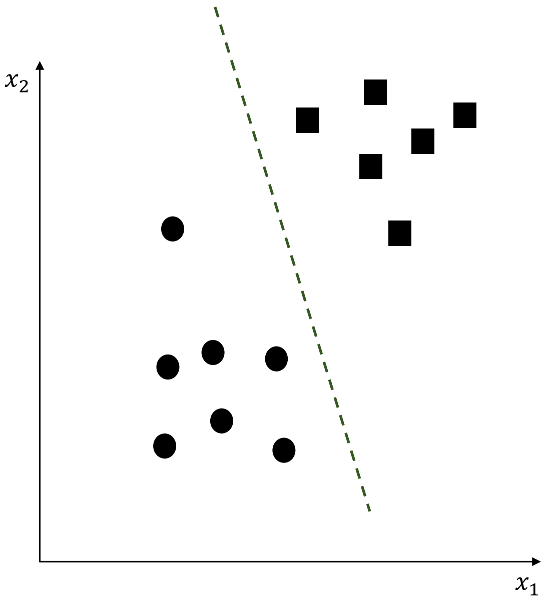


Figure 3: Schematization of a binary 2-D classification problem.

The above is a classic example of a two-dimensional linear classifier, that is a classifier that separates a set of objects into their respective groups with a line. Most classification tasks, however, are not that simple, and often more complex structures are needed in order to make an optimal separation. Support Vector Machines make use of a set of mathematical functions, known as kernels[1], to rearrange the objects (i.e. to perform a mapping).

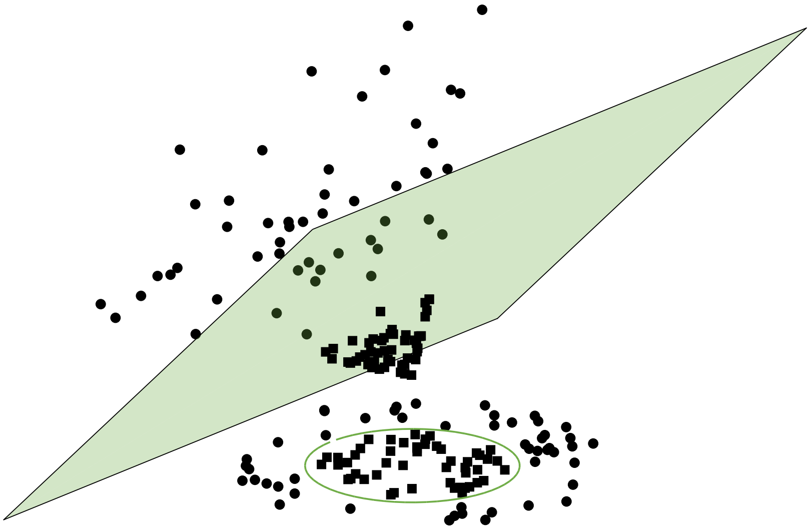


Figure 4: Non-linear mapping. In this example, the 2D feature vectors are mapped through a 2^{nd}-degree homogeneous polynomial kernel

Referring to Figure 4, the mapped objects are linearly separable. Thus, instead of directly building the complex curve, what is done is to find an optimal hyperplane (when more than two dimensions are considered), in the new object space, that can separate the two groups. Following the description in [27], a more detailed look on the so-called C-SVM formulation for classification and ϵ-SVM for regression is hereby given.

2.2 C-SUPPORT VECTOR CLASSIFICATION

A classification task is concerned with identifying to which of a set of categories a new observation belongs, on the basis of some known data. Using the appropriate terminology, each *instance* in the dataset belongs to a certain *class* and depends on several *features*. Assuming that each object to be classified or processed somehow can be represented as a fixed-size feature vector, typically of the form $\mathbf{x} \in \mathbb{R}^n$, the idea is to derive a linear discriminant function which maximizes the *margin*, i.e. the perpendicular distance to the closest point, as shown in Figure 5.

Aside from the large margin principle, a probabilistic interpretation of SVM is also given in [27]

Given a set of training vectors $\mathbf{x}_i \in \mathbb{R}^n$, $i = 1, \ldots, l$ that belongs to two classes, identified by the class labels $y_i \in \{1, -1\}$, the support

[1] A kernel is a similarity function. Under some conditions (Mercer's theorem), it can be defined as a real-valued function of two arguments $K(x, y) = \phi(x) \cdot \phi(y)$

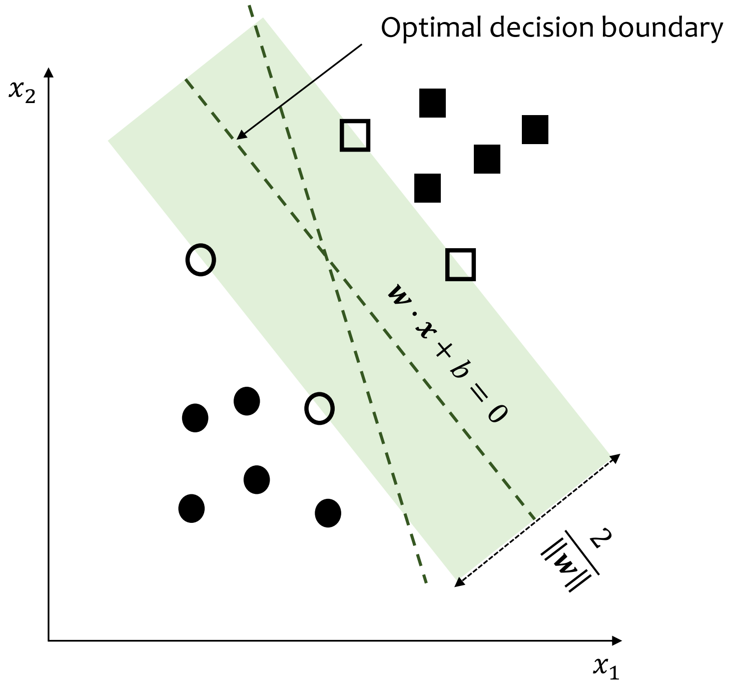

Optimal decision boundary
x_2
x_1
$\boldsymbol{w}\cdot\boldsymbol{x}+b=0$
$\frac{2}{\|\boldsymbol{w}\|}$

Figure 5: Geometry of a linear decision boundary in 2-D. The support vectors are the empty dots and squares.

vector classifier solves a standard Quadratic Program (QP), as given by:

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\xi_i \\ \text{subject to} \quad & y_i(\mathbf{w}^T\phi(\mathbf{x}_i)+b) \geqslant 1-\xi_i \\ & \xi_i \geqslant 0 \end{aligned} \tag{1}$$

where $C > 0$ is a regularization parameter that controls the number of errors that are tolerated on the training set (thus, closely related to overfitting), $\phi(\mathbf{x}_i)$ is some function that maps the training vectors into a higher-dimensional space, so that a linear model can still be used in the presence of non-linear data, and ξ_i are slack variables such that $\xi_i = 0$ if the point is on or inside the correct margin boundary, or $\xi_i = |y_i - f(\mathbf{x}_i)|$ otherwise.

For computational efficiency, what is done is to eliminate the primal variables $\mathbf{w}$, w_0 and ξ_i, and just solve the dual variables, which correspond to the Lagrange's multipliers for the constraints. The optimization problem's dual form reads as follows:

$$\begin{aligned} \max_{\lambda} \quad & -\tfrac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)\lambda_i\lambda_j + \sum_{i=1}^{N}\lambda_i \\ \text{subject to} \quad & \sum_{i=1}^{N}\lambda_i y_i = 0 \\ & 0 \leqslant \lambda_i \leqslant C \end{aligned} \tag{2}$$

where λ_i are Lagrange's multipliers. The advantages are the followings:

- Simpler constraint equations
- The training vectors appear as mutual scalar products. This is of fundamental importance to apply the so-called kernel trick (i.e., making the $K(x, y) = \phi(x) \cdot \phi(y)$ substitution). In this way, the feature mapping is performed without blowing up dimensionality.
- The optimal solution depends only on a sub-set of the training vectors (the *support vectors*), those with non-zero Lagrange's multipliers.

Specialized algorithms, which avoid the use of generic QP solvers, have been developed for this kind of problem, such as the Sequential Minimal Optimization (SMO) algorithm. After problem (2) is solved, using the primal-dual relationship, the optimal **w** satisfies:

$$\mathbf{w} = \sum_{i=1}^{N} y_i \lambda_i \phi(\mathbf{x}_i) \tag{3}$$

and the decision function is given by:

$$\operatorname{sgn}(\mathbf{w}^T \phi(\mathbf{x}) + b) = \operatorname{sgn}\left(\sum_{i=1}^{N} y_i \lambda_i K(\mathbf{x}_i, \mathbf{x}) + b\right) \tag{4}$$

For the sake of clarity, a brief list of the most popular kernel functions is provided in Table 1. With the exception of the linear kernel, which may be the only feasible choice for extremely large, high-dimensional datasets (from a computational point of view), the choice of the kernel usually depends on a-priori knowledge about the dataset. However, it is well established in the literature how the Gaussian RBF kernel is generally more flexible and well-performing than the others, because ideally one can model any arbitrary continuous function with its infinite-dimensional (*implicit*) function space.

For the RBF kernel, one can also define $\gamma = 1/2\sigma^2$

Table 1: Popular choices for the Kernel function

Name	Function
Linear	$\mathbf{x}_i^T \mathbf{x}_j$
Polynomial	$(a + \mathbf{x}_i^T \mathbf{x}_j)^b$
Gaussian Radial Basis Function	$\exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$

2.2.1 *Multi-Class Classification*

Support Vector Machines were originally designed for binary classification, but some methods are available to extend them to multi-class. A valid approach is to consider a set of binary classifiers. It requires a coding design, which determines the classes that the binary learners train on, and a decoding scheme, which determines how the results (predictions) of the binary classifiers are aggregated. In this work, the so-called one-vs-one approach is used. It requires $k(k-1)/2$ binary learners (k = number of classes) and, for each binary learner, one class is positive, another is negative, and the rest are ignored. A voting strategy is adopted to determine the class of a new observation. This design exhausts all combinations of class pair assignments. Hsu and Lin [19] prove in their study how this approach generally yields the best performance when compared to other schemes.

2.3 training & validation

A simple, introductory article on the topic can be found on [1]

When building a machine learning model, it is absolutely necessary to assess the stability and performance of said model over unseen data, before releasing it as the final product. It cannot just be fit to the training data, hoping it would accurately work for the real ones. Some kind of assurance is needed, that your model has got most of the patterns from the data correct, and its not picking up too much on the noise, i.e. *overfitting* the training data. The process of deciding whether the numerical results quantifying hypothesized relationships between variables are acceptable as descriptions of the data, is known as validation. Generally, an error estimation for the model is made after training, better known as evaluation of residuals. In this process, a numerical estimate of the difference in predicted and original responses is done, also called the training error. Different metrics can be used in this regard, such as number of instances correctly classified for classification tasks or RMSE for regression (it largely depends on the problem at hand and on the properties of characterizing dataset). However, this only gives an idea about how well the model does on data used to train it. Therefore, in order to improve the *generalization* performance of the machine learning model, a validation procedure using different data than the training ones is performed.

A brief and general introduction to the most common validation procedures is hereby given

2.3.1 *Holdout Validation*

A basic method consists of removing a part of the training data and using it to get predictions from the model trained on rest of the data. The error estimation then tells how the model is doing on unseen

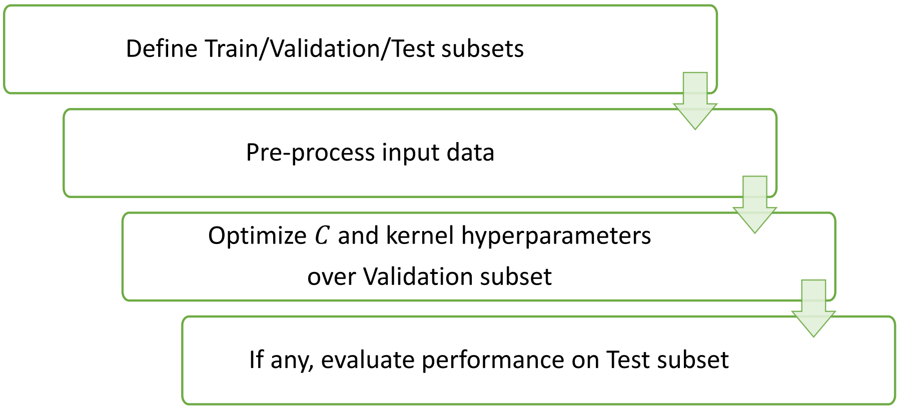

Define Train/Validation/Test subsets
Pre-process input data
Optimize C and kernel hyperparameters over Validation subset
If any, evaluate performance on Test subset

Figure 6: Schematization of the validation process for SVM. Data pre-processing includes, but it is not limited to, cleaning, instance/feature selection, normalization, transformation

data, i.e. the validation set. This is a simple kind of validation technique, also known as the holdout method. Although this method doesn't take any overhead to compute it still suffers from issues of high variance. This is because it's not straightforward to select which data points to include in the validation set and the result might be entirely different for different sets.

2.3.2 *k-Fold Cross-Validation*

Typically, one searches for a method that provides ample data for training the model and also leaves ample data for validation. k-Fold cross-validation tries to address this very issue. A generalization of the Holdout method, k-Fold cross-validation splits the data into K subsets. Now the Holdout method is repeated k times, such that each time, one of the k subsets is used as the test set/ validation set and the other k-1 subsets are put together to form a training set. The error estimation is averaged over all k trials to get the total effectiveness of the model. As can be seen, every data point gets to be in a validation set exactly once, and gets to be in a training set k-1 times. This significantly reduces bias as we are using most of the data for fitting, and also significantly reduces variance as most of the data is also being used in validation set. Interchanging the training and test sets also adds to the effectiveness of this method. As a general rule and from empirical evidence, k = 5 or 10 is generally preferred. k-Fold cross-validation can be taken to an extreme with Leave-One-Out validation, where only a single instance is left out.

2.4 PREPROCESSING THE DATA

Machine learning algorithms learn from data. It is critical to feed them the right data for the problem under consideration and to make sure that they are in a useful scale, format and even that meaningful features are included. The entirety of the process can be summarized with the following steps.

1 Data Selection. Consider what data is available, what data is missing and what data can be removed.

2 Data Preprocessing. Organize the selected data by formatting, cleaning and sampling from it. Sampling training data is fundamental when dealing with imbalanced dataset. The approach considered in this work is the ADASYN sampling approach for learning from imbalanced data sets. The essential idea of ADASYN is to use a weighted distribution for different minority class examples according to their level of difficulty in learning, where more synthetic data is generated for minority class examples that are harder to learn compared to those minority examples that are easier to learn. As a result, the ADASYN approach improves learning with respect to the data distributions in two ways: by reducing the bias introduced by the class imbalance, and by adaptively shifting the classification decision boundary toward the difficult examples [18].

3 Data Transformation. Transform preprocessed data ready for machine learning by engineering features using scaling, attribute decomposition and attribute aggregation. Standardization of predictors using the corresponding mean and standard deviation or normalization over a fixed interval are common choices for data scaling.

2.5 ALGORITHM IMPLEMENTATION

The Support Vector Machine implementation from MATLAB® is used in the present work. In addition, the results have been compared with the LIBSVM [10] software package, achieving comparable performance.

Part II

ANOMALY DETECTION WITH APPLICATION TO PHILAE SOLAR GENERATOR

3 ON THE NEED OF AUTONOMOUS FAULT DETECTION

This chapter presents the first application case of the thesis, describing the Rosetta and Philae mission. Then, a description of the modeling assumptions and procedure is provided.

3.1 PROBLEM OVERVIEW

Adequate and timely reaction of the spacecraft to changes in its operational environment, as well as in the operational status of the system, are key aspects in the field of autonomous operations. Classical Fault Detection, Identification and Recovery (FDIR) approaches suffer from the problem of deferring decisions to the ground segment: they indeed represent a reactive approach, that cannot provide and utilize prognosis for the imminent failures. On the other hand, spacecraft multidimensional telemetry data can contain a wealth of information about complex system behavior and they indeed represent a valuable resource to build a robust and effective fault detection model. Machine learning and data mining algorithms can be used to examine and extract information from archived or even simulated data (Gao et al. [16], Zhao et al. [31]).

This part of the thesis is concerned with the application of a Support Vector Machine classifier to successfully tackle the problem of identifying anomalies in the solar generator subsystem of an interplanetary probe, namely the Rosetta's lander Philae, for which the number of sensors' readings is scarce and an autonomous fault detection model is advisable. Learning from the experience of the engineers who worked in the ground stations during lander deploying and descent, it is evident how any kind of autonomous decision-making (even only as back-up) would have been highly favorable. A variety of critical issues arose, such as communication time lag (signal could take one hour to travel back and forth), uncertainty of communication time windows and grossly unknown landing environment. As a final note, the choice of the case study falls on the solar generator subsystem because of its vital importance and possibility to draw a parallel with existing applications in ground solar plants.

The remaining of the chapter is therefore dedicated to briefly introduce at first the mission and then describe the modeling choices.

3.2 THE REAL SYSTEM: ROSETTA AND PHILAE

3.2.1 *Introduction*

Approved in 1993 as a Cornerstone Mission in ESA's Horizons 2000 Science Programme and launched in March 2004, Rosetta represents one of the most succesful european space missions. The scientific objective was to perform an accurate study of comet 67P/Churyumov-Gerasimenko, both from range and up-close, with the aid of the lander Philae. Rosetta is also the first deep space mission to rely exclusively on solar arrays for power generation. The delivery of Philae took place on November 12th, 2014 at a distance of about 3 AU from the Sun, where the solar intensity is 150 W/m^2 ca.

The solar intensity at Earth mean distance from the Sun is 1366 $\frac{W}{m^2}$ ca. and then adjusted using the inverse square law

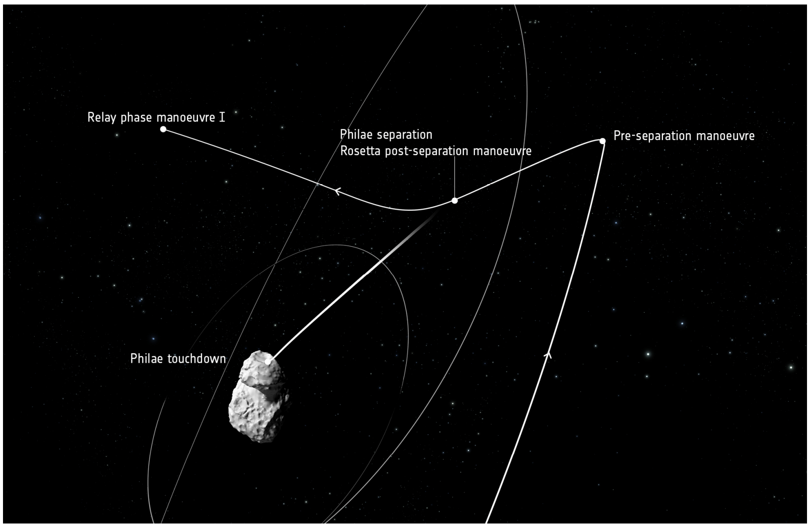


Figure 7: Trajectory of Rosetta's orbit, focusing on the manoeuvres on 12 November 2014. ©ESA

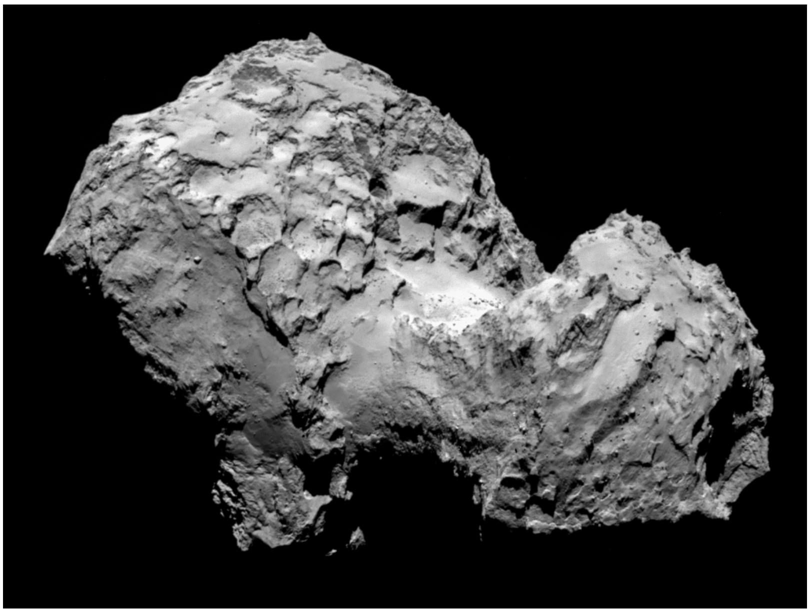
67P/67P/Churyumhuryumov-Gerasiv-Gerasimenkonko is a short period comet of high scientific value, being a remnant of the Kuiper belt. It was discovered in 1969 by Ukrainian astronomers Klim Churyumov and Svetlana Gerasimenko. It is characterized by a high eccentric orbit with a low inclination: as a reference, its orbital parameters are listed in Table 2. After the arrival of Rosetta in August 2014, a detailed survey allowed to discover the peculiar duck-like shape and the spin parameters (see Figure 8).

3.2.2 *Lander Philae*

From the thermal point of view, the environmental conditions in which the lander operated changed extremely during its mission. Philae was exposed for about 6 hours to the irradiation coming from the Sun, and in the next 6 hours it dissipated heat towards the cold space. The temperature ranged from -80 °C to 20 °C during the day, going

Table 2: Comet 67P Orbital Parameters [29]

Orbital Parameter	Units	Symbol	Value
Semi-major axis	[AU]	a	3.464312068995
Eccentricity	[-]	e	0.641099280875
Inclination	[deg]	i	7.045643228574
Longitude of the asc. node	[deg]	Ω	50.177077915429
Argument of pericentre	[deg]	ω	12.711342250349



(a)

Parameter	Units	Symbol	Value
Rotation Period	[h]	T_{rot}	12.76
Axis right ascension	[deg]	α_{sa}	69.30
Axis declination	[deg]	β_{sa}	64.10
Obliquity	[deg]	ϵ	52.00

(b)

Figure 8: Comet 67P on Aug 3, 2014 and its properties

below -200 °C during the night at the beginning of the mission. Such a large temperature variation has strong influence on the performances of its solar generator and its subsystems. A second dominating factor is the distance of the comet from the Sun, due to the large eccentricity of the comet's orbit. At the beginning, Philae was nearly at 3 AU and the irradiation was very low, increasing up to its maximum value at perihelion (1.2 AU ca.). This effect is enhanced by the seasonal cycle that is caused by the spin axis inclination, therefore there are surfaces that receive higher insolation and, on the other hand, places that are subjected to longer periods in shadow. In order to satisfy Philae's operative thermal range (from -50 °C to +60 °C [29]), the system was separated into three main structures:

- Warm Compartment: it is designed to store the scientic tools and the devices that are sensitive to extreme temperature values

- Cold Balcony: the external shell, which permits interaction with the environment for scientific experiments and acts as a support for the Solar Hood.

- Solar Hood: the solar cell generator, made of six panels where the solar cells are assembled. Built by DLR, was made of six carbon-fiber sandwich panels (for a total area of about 2 m^2). It consisted of a unique structure collecting the following panels: Wall 1, Wall 2, Wall 3, Wall 4, Wall 5 and the lid, and by two detachable panels, identified as Balcony 1 and Balcony 2, which were attached by means of a connector to Wall 1 and Wall 5, respectively (Figure 9).

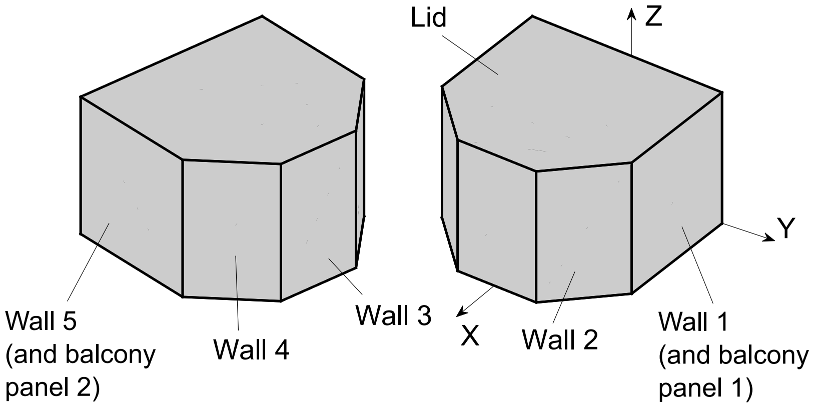


Figure 9: Philae Solar Array configuration

Thermal insulation was also provided by two MLI tents which separate the Warm Compartment from the Cold Balcony and the Solar Hood. In addition, there were two solar absorbers and hibernation heaters.

During the comet in-situ investigations, the lander Solar Arrays (SA) were designed to produce about 8 W. They were composed of 1224 10LiTHI-ETAR 3-ID/200 silicon solar cells, organized in six arrays, as mentioned above. These cells were the same of the Rosetta orbiter but with different dimensions (32.4 mm x 33.7 mm, 200 µm thick, for the lander) and coverglass applied [30]. They were produced in the frame of ESA R&D program "Solar cells for Low Intensity and Low Temperature (LILT) operations" in order to avoid LILT degradation effects. As shown in Figure 10, the architecture chosen for the electrical network make use of Maximum Power Point Trackers: this is true for nominal operations of the lander, and indeed this will be the case hereby considered. Power distribution was performed via the Lander Primary Bus (non-stabilized, +28 V baseline [8]).

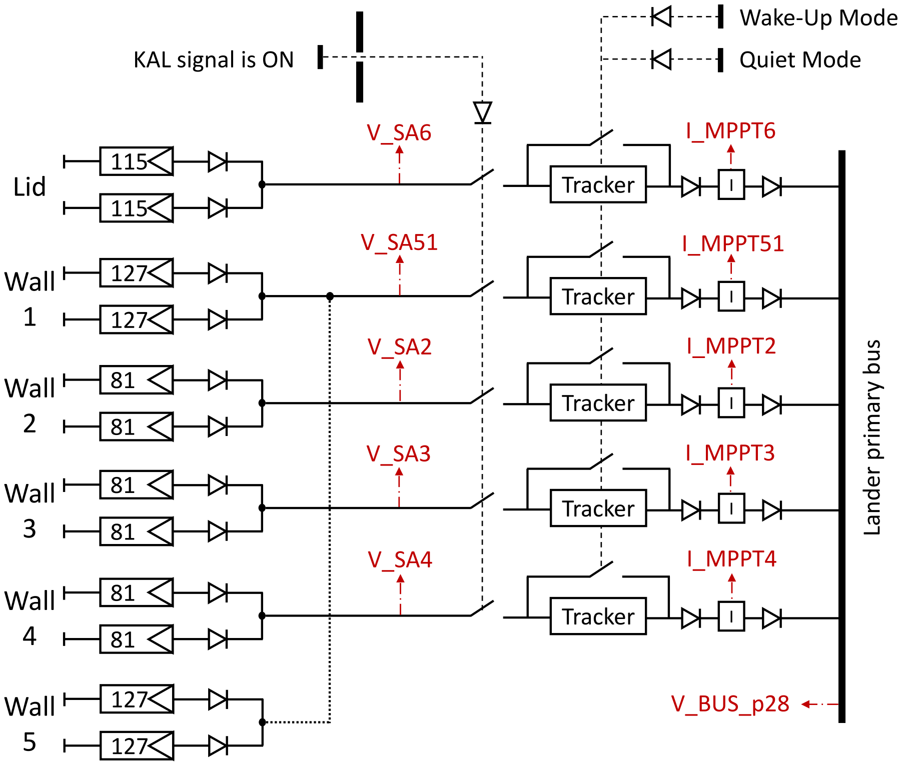


Figure 10: Philae Solar Generator Electrical Scheme

3.3 MODELING APPROACH

In order to devise an intelligent fault detection algorithm, a multidisciplinary approach has been taken to properly model the real system, i.e. how the telemetry is produced. In the following, assumptions and procedure are described.

3.3.1 *Comet 67P Churyumov-Gerasimenko and Sun Path*

The main assumptions adopted for the modeling of comet 67P are summarized in Table 3. Here the driver is to produce a preliminary estimation of the Sun path (in terms of Azimuth and Elevation angles) with respect to Philae. At the beginning of the mission, the comet is assumed to be sufficiently far from the Sun to be considered almost inactive, so that light scattering and attenuation due to sublimated particles are negligible [29]. As a consequence, a direct illumination model is considered. In addition, the cometary albedo is low enough

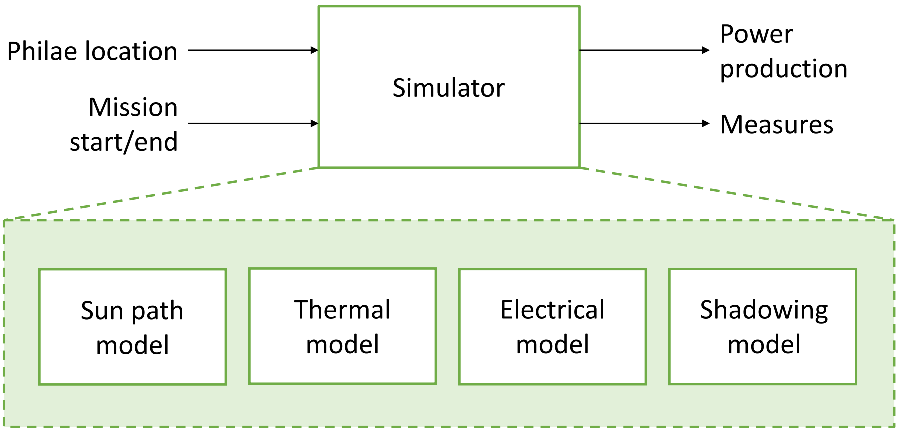


Figure 11: Philae Simulator block scheme

to be neglected at first [29] and the inclination of the lander with respect the local normal is assumed to be zero.

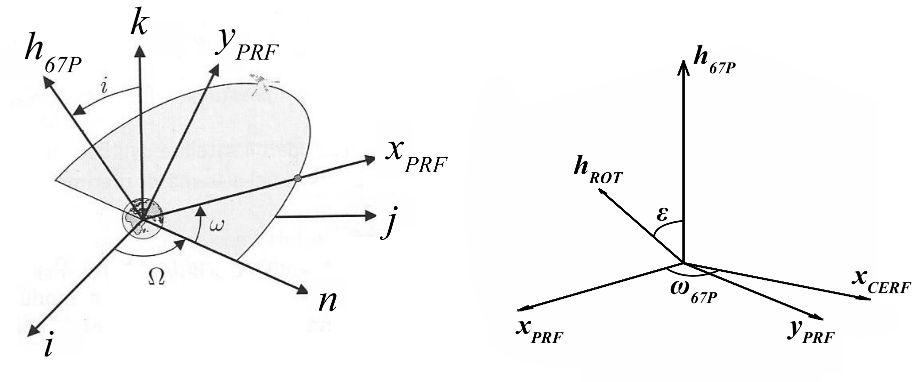
Table 3: Comet 67P Modeling Assumptions

Feature	Assumption
Shape	Spherical shape
Kinematics	Fixed spin axis, constant angular velocity
Orbit	Defined by Keplerian parameters, no perturbations
Ref. Frames	Interconnected orthogonal rotation matrices

Some details on the Sun Path estimation are provided in the following. Three different reference systems are considered. The first one is the Ecliptic Reference System (ERF), whose x-axis points towards the Earth vernal equinox and whose z-axis coincides with the Earth angular momentum $\mathbf{h}_{\oplus}$. The second one is the Perifocal Reference Frame (PRF), whose x-axis points towards the perihelion of the comet and whose z-axis coincides with the comet angular momentum $\mathbf{h}_{67P}$. The last reference system is the Comet Equatorial Reference frame (CERF), whose x-axis belongs to the intersection between orbital and equatorial plane of Churyumov-Gerasimenko and the z-axis coincides with the spin axis of the comet. Considering the orbital parameters given in Table 2 and using the rotation matrix in (5), it is possible to move from ERF to PRF. This matrix collects three different rotations: the first one is a rotation around the z-axis of an angle ω, the second one is performed around the nodal axis $\mathbf{n}$ of an angle i, while the last one is around $\mathbf{h}_{67P}$ axis of a angle ω. Figure 12a shows the three rotations.

$$R_{ERF\to PRF} = \begin{bmatrix} c\Omega c\omega - s\Omega s\omega ci & s\Omega c\omega + c\Omega s\omega ci & s\omega si \\ -c\Omega s\omega - s\Omega c\omega ci & -s\Omega s\omega + c\Omega c\omega ci & c\omega si \\ s\Omega si & -c\Omega si & ci \end{bmatrix} \quad (5)$$

The x-axis of the CERF is identified by the following cross-product:



(a) From ERF to PRF [26]

(b) From PRF to CERF

Figure 12: Coordinate transformations

$$x_{\text{CERF}} = \mathbf{h}_{67P} \times \mathbf{h}_{\text{rot}} \tag{6}$$

where $\mathbf{h}_{\text{rot}}$ is the comet spin axis. Once the angle ω_{67P} is determined, a rotation matrix from PRF to CERF can be written. The matrix expression is reported in (7), while Figure 12b shows the two rotations.

$$R_{\text{CERF}\rightarrow\text{PRF}} = \begin{bmatrix} c\omega_{67P} & s\omega_{67P}c\epsilon & s\omega_{67P}s\epsilon \\ -s\omega_{67P} & c\omega_{67P}c\epsilon & c\omega_{67P}s\epsilon \\ 0 & -s\epsilon & c\epsilon \end{bmatrix} \tag{7}$$

It is now possible to express the position vector Sun-Comet in the CERF and thus computing the Sun declination angle β_{Sun}. The expression is showed in Equation 8, where $\hat{k}_{\odot-67P}$ is the z-component of the unit vector $\hat{r}_{\odot-67P}$.

$$\beta_{\text{Sun}} = \arcsin(\hat{k}_{\odot-67P}) \tag{8}$$

The comet position at a given julian date (JD) has been retrieved knowing the JD of the passage at the perihelion (JD_p = 2457247.5908) and solving the Kepler's equation (9) for E, the eccentric anomaly.

$$M = E - e\sin E \tag{9}$$

M represents the mean anomaly and it is defined as $M = \frac{T}{2\pi}(t - t_p)$. The trascendental equation has been solved using the Matlab function *fzero*. The initial guess E_0, as suggested by Mengali and Quarta [26], is reported in Equation 10.

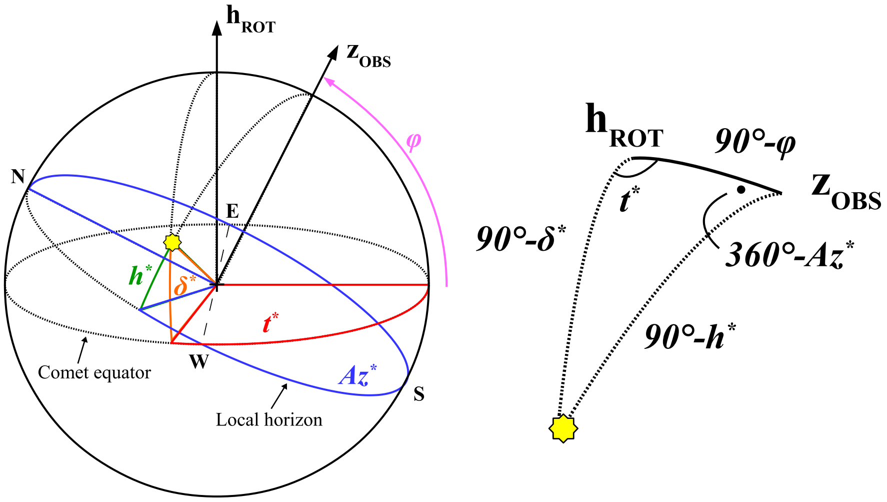
$$E_0 = M + e\left[\sqrt[3]{\pi^2 M} - \frac{\pi}{15}\sin M - M\right] \tag{10}$$

Once the transcendental equation has been solved numerically, the comet true anomaly can be obtained as per Equation 11:

$$\theta = 2\arctan\left(\sqrt{\frac{1+e}{1-e}}\tan\frac{E}{2}\right) \tag{11}$$

Knowing θ and the other five orbital parameters, the state vector in terms of position and velocity is easily evaluated.

Finally, in order to uniquely compute Azimuth and Elevation of the Sun at a given day, latitude and time angle, let us consider the spherical triangle in Figure 13b:



(a) Comet celestial sphere

(b) Spherical triangle

Figure 13: Spherical geometry

where

- $\mathbf{h}_{\text{rot}}$: 67P spin axis
- $\mathbf{z}_{\text{obs}}$: Philae zenith
- $t_{\odot}$: time angle, measured on the celestial equator between the observer and the Sun
- $\delta_{\odot}$: Sun declination
- $Az_{\odot}$: azimuth, measured eastward on the horizon starting from North
- ϕ: Philae latitude

During a cometary day, the time angle goes from 0°to 360°. Knowing $t_{\odot}$ and using sine and cosine theorems, the azimuth and the altitude of the Sun at a selected day and comet latitude can be computed. Since $h_{\odot} \in [-90°, 90°]$, Equation 12 gives the altitude with no indetermination.

$$h_{\odot} = \arcsin\left(\sin\delta_{\odot}\sin\phi + \cos\delta_{\odot}\cos\phi\cos t_{\odot}\right) \tag{12}$$

Instead, the azimuth is given by:

$$\begin{aligned} \sin Az_{\odot} &= -\sin t_{\odot}\cos\delta_{\odot} / \cos h_{\odot} \\ \cos Az_{\odot} &= \left(\sin\delta_{\odot} - \sin h_{\odot}\sin\phi\right) / \left(\cos h_{\odot}\cos\phi\right) \\ Az_{\odot} &= \arctan\left(\sin Az_{\odot} / \cos Az_{\odot}\right) \end{aligned} \tag{13}$$

3.3.2 *Electrical Model of the Solar Cell*

A solar cell is an electronic device which directly converts sunlight into electricity. Light shining on the solar cell produces both a current and a voltage to generate electric power. Typically, the efficiency increases for decreasing temperatures until -50 °C and then fall at lower temperatures. The output current of solar cells falls linearly with illumination intensity. Solar cells have a non-trivial relation between solar irradiation, temperature and total resistance that produces a non-linear output, which is usually depicted by the I-V curve, where I is the current and V is the voltage of the cell. To begin with, it is useful to identify a few fundamental parameters:

- *Short-Circuit current* I_{SC}: the short-circuit current is the current through the solar cell when the voltage across the cell is zero (i.e., when the solar cell is short circuited). It is, ideally, the largest current which may be drawn from the solar cell.

- *Open-Circuit voltage* V_{OC}: the open-circuit voltage is the maximum voltage available from a solar cell, occurring at zero current.

- *Maximum power current and voltage* V_{MP}, I_{MP}: the max power current/voltage are the values relative to the maximum amount of generated power ($P = V \cdot I$).

- *Efficiency* η: it quantifies how much of the incident power on the surface is converted into electrical energy. It is defined as $\eta = (V_{MP} \cdot I_{MP})/P_{inc}$.

In this work, the voltage-current relation for the single solar cell is given by a modified Shockley model, that takes into account the negative voltage region of the I-V curve. This is of extreme importance because Philae solar arrays do not have by-pass diodes [17]. The following equations describe the model (see [17] for additional details):

$$I(V) = \begin{cases} I_{PV,adj} - B\left(e^{\frac{V}{C}} - 1\right) & \text{if } V \geqslant 0 \\ I_{PV,adj} + I_d(V) & \text{if } V < 0 \end{cases} \tag{14}$$

where

$$I_d(V) = \begin{cases} 0 & \text{if } V \geqslant V_{od} \\ -\frac{V - V_{od}}{R_d} & \text{if } V < V_{od} \end{cases} \tag{15}$$

and

- V, I are the voltage and current.

- $I_{PV,adj}$ is the value of the short-circuit current I_{sc}, that accounts for eventual cell malfunctioning, light incidence angle, radiation damage, cell temperature and dust ratio (according to [30], [9], [17]).
- B, C are coefficients function of the four main electrical parameters (voltage and current at maximum power point, open-circuit voltage, short-circuit current).
- V_{od} is the intrinsic diode voltage drop at reverse biasing and R_d is the intrinsic diode resistance. They are both non-linear function of the cell temperature.

Finally, knowing that the solar cells configuration is as reported in Table 4, a scaling process to string/array level is implemented.

Table 4: Philae SA sections layout

Solar Array	Electric Section	String in Array	Cells in String
SA1	Wall 1 + Balcony 1	2	127
SA2	Wall 2	2	81
SA3	Wall 3	2	81
SA4	Wall 4	2	81
SA5	Wall 5 + Balcony 2	2	127
SA6	Lid	2	115

3.3.3 *Thermal Model of the Solar Cell*

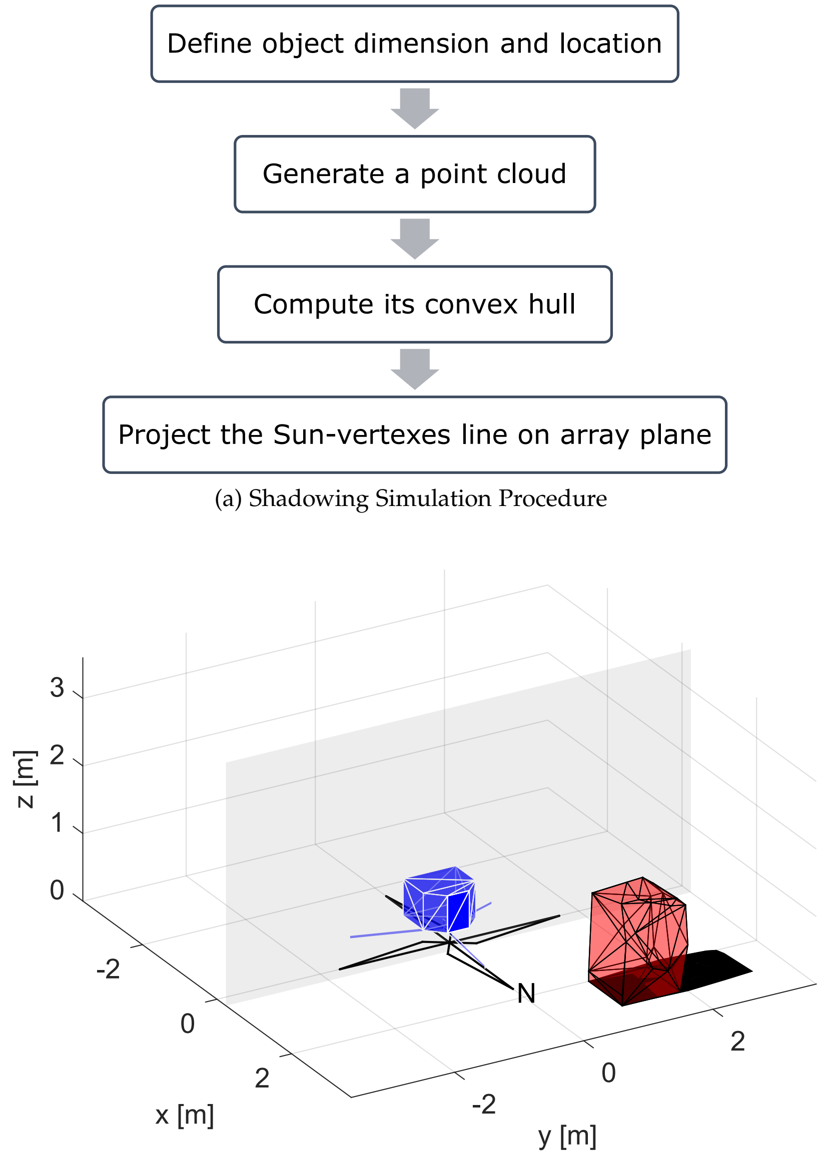
Previous works deal with the modeling of the thermal behavior of Philae solar arrays in different ways: in [8], an analytical formula that expresses the temperature as a function of the solar irradiance only is used, while in [29], the temperature of each panel is assumed constant. An improvement is made in this work. A non-stationary lumped parameter model is considered, with six independent nodes, one for each array. The panel temperature is assumed uniform throughout the constitutive layers. The governing equations are given by:

$$C_i \frac{dT_i}{dt} = Q_{in,i} - Q_{out,i} \,, \qquad i = 1, \dots, 6 \tag{16}$$

where

- $C_i = \sum_m A_i h_m \rho_m c_m$ is the thermal capacity of Philae i-th solar panel, assumed to be made by the contribution of m different layers [5].
- Q_{in} is the in-going thermal power contribution given by the Sun irradiance and internal heat generation [29].

- Q_{out} is the out-going thermal power that accounts for heat exchanged by radiation with the deep space and the comet surface and for the thermal energy taken from the module in the form of the electrical energy generated [21].


Define object dimension and location
Generate a point cloud
Compute its convex hull
Project the Sun-vertexes line on array plane
z [m]
3
2
1
0
-2
0
2
x [m]
N
-2
0
2
y [m]

(a) Shadowing Simulation Procedure

(b) Example

Figure 14: Array periodic shadowing. Panel no. 3 is taken as a reference

3.3.4 *Panel shadowing*

A main drawback of the simplifications made in Section 3.3.1 is that panel shadowing due to irregularities of the landing site (as it was the case in the actual scenario) is not taken into account. Static performance decrease can be included within the solar array model previously described, by properly tuning the input parameters for each cell. A dynamic shadowing condition, instead, results in periodic oscillations of the power output according to the day-night cycle, and must not be considered as a faulty state. The effect is simulated by creating a physical obstacle in the neighborhood of Philae. Knowing the Sun path, the obstacle shadow is the projection of the sun-vertexes line onto the array plane. The intersection between the area of the projected shadow and the solar array quantifies how much the whole panel/single cell is shadowed (see Figure 14). This method is fairly general and eventually allows for the creation of different obstacles shapes with ease.

3.3.5 *Choice of the SVM Model Features*

In the case of an already existing system, the choice of representative features, worth to be included in the predictive model, is constrained by the available telemetry. From the electrical scheme in Figure 10, it appears that the available telemetry consists of the arrays voltage and the MPPT output current, as the bus is set to 28 V. The actual value of these two parameters is strictly related to the MPPT architecture. Since the available information are not sufficient to develop a model able to produce results for wide operational ranges [9], it has been decided to capture instead the average behavior of the system. A constant MPPT efficiency is introduced (as in [29]), so that the generated power is a fraction of the maximum available power. Then the output current is computed as the ratio between the power generated and the bus voltage [8]. Finally, without any loss of generality, the voltage is assumed to be equal to the open-circuit voltage: indeed, even if the values of real telemetry output voltage and V_{oc} are different, they exhibit the same dependance from environmental parameters. As shown in Chapter 4, a successful classification process requires at least a third attribute, assumed to be the sun incidence angle, i.e. the angle between the panel normal and sun direction. A method to compute this angle, starting from the lander current telemetry and the peculiar physical shape, is analyzed in Caputo [8].

4 ANALYSIS OF THE RESULTS - PART I

The chapter presents the results of the study for what regards the ability of a Machine Learning model, built with the available telemetry on Philae, to correctly classify abnormal from nominal behavior of the Solar Generator subsystem. The analysis is restricted to situations for which the power output of the array is less then what expected. Also, the analysis is referred to panel no. 3 (see Figure 9), but results and considerations are equivalent for the other five arrays. Indeed, because of the assumption of separate thermal nodes in Section 3.3.3, each array is functionally decoupled from the others. At first, telemetry values under arbitrarily varying temperature and incidence angle are simulated. Reference irradiance is the value at 3 AU (beginning of mission) and radiation damage is accounted for. The aim is to reproduce an a-priori test campaign of the array to obtain training data.

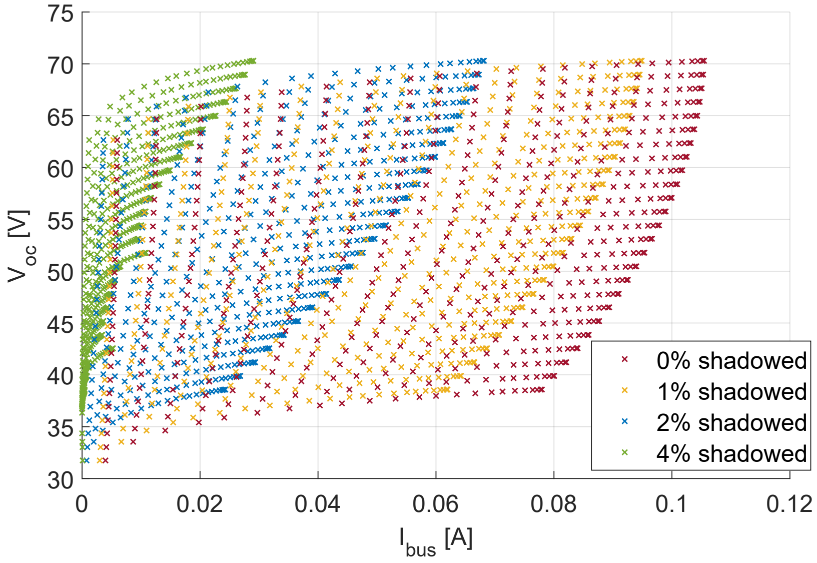


Figure 15: Current and Voltage plot under varying environmental conditions

Figure 15 shows the outcome of the simulation. Without further investigation, it is clear that separating nominal against faulty conditions merely by the two-dimensional dataset is impossible (similar to what can be found in the work by Zhao et al. [31]). Data points belonging to different classes overlap and no separating hyperplane would be able to produce satisfactory predictions. Therefore, a third attribute needs to be used to detect and classify the faults. As anticipated in Section 3.3.5, the choice falls upon the sun incidence angle. Figure 16 depicts the three-dimensional training set.

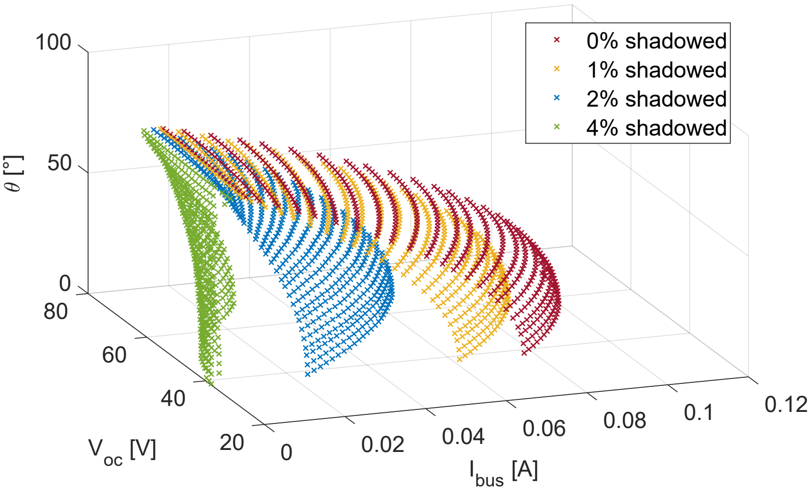
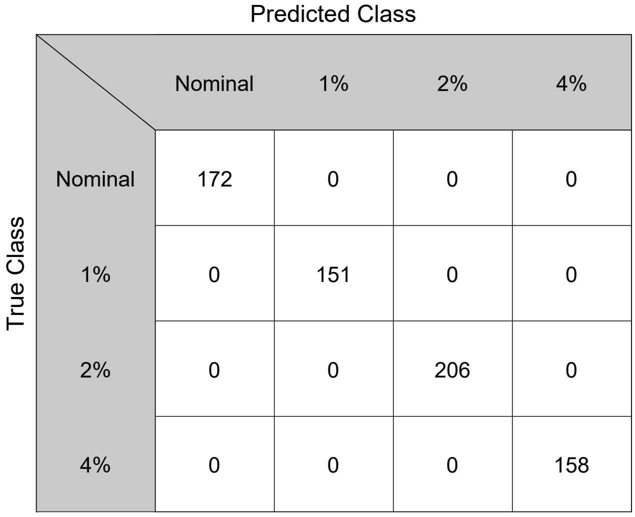


Figure 16: Current, Voltage and Incidence Angle plot under varying environmental conditions

As expected, it can be noticed how the output power, related to I_{bus}, decreases very rapidly as the number of inoperative cells grows. This result is coherent with the studies performed in Cattafesta [9], Gaspari [17]. The minimum set of attributes is then defined as the MPPT output current I_{bus}, the open-circuit voltage V_{oc} and the sun incidence angle θ. The training process can be initialized: the data from Figure 16 are split into training and testing sets, and normalized in the $[0, 1]$ range. A Radial Basis Function kernel is used and the hyperparameters $[C, \gamma]$, which define the shape of the hyperplane and therefore the classification performances, are optimized through a grid-search algorithm and 10-fold cross-validation. The confusion matrix is reported in Figure 17.


Predicted Class

True Class

	Nominal	1%	2%	4%
Nominal	172	0	0	0
1%	0	151	0	0
2%	0	0	206	0
4%	0	0	0	158

Figure 17: Confusion Matrix, case # 1. Percentages refer to no. of inactive cells.

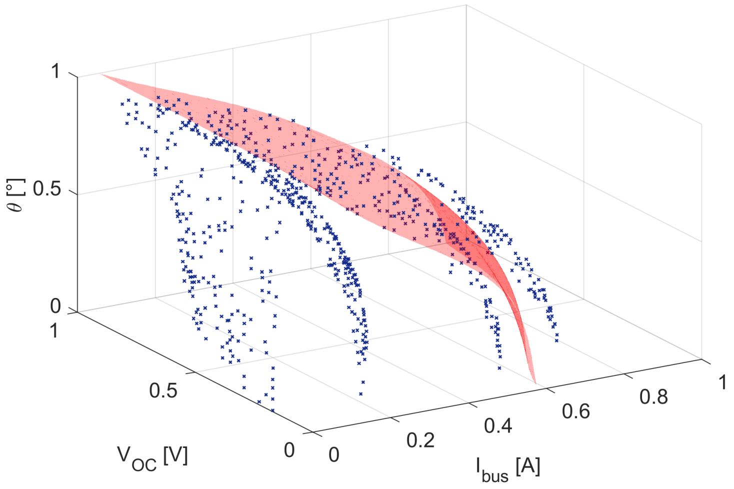


Figure 18: Classification Hyperplane Example (normalized inputs)]

A confusion matrix is a table that is often used to describe the performance of a classification model on a set of test data for which the true values are known. It is clear how the SVM model is able to perfectly distinguish between various degrees of static shadowing, equivalent to permanently broken cells.

Any damage or malfunction on the solar cells of the lander (or similar space probe), which directly results in less-than-optimal power output, can be then properly captured. A second test is then performed. Simulated telemetry is produced over one cometary day, for nominal and faulty cases. The results are again satisfactory as the data points fall into the training region, as shown in Figure 19 for the nominal case. The zero I-V values correspond to fully shaded conditions, due to day-night cycle.

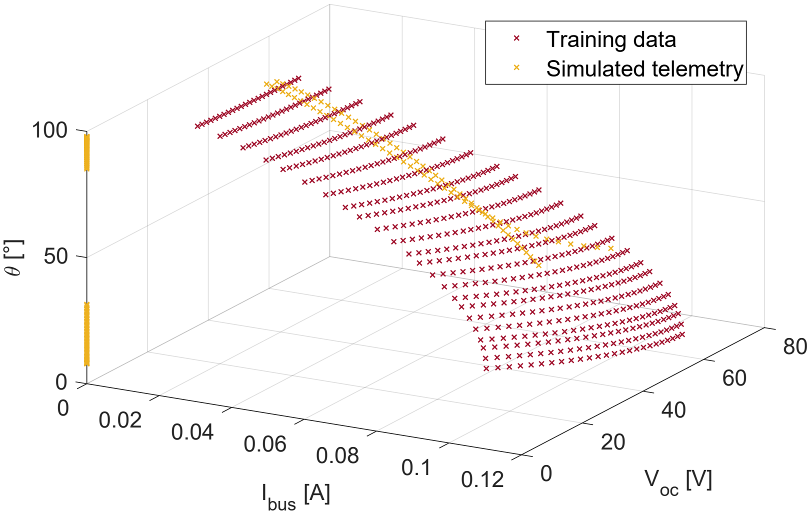
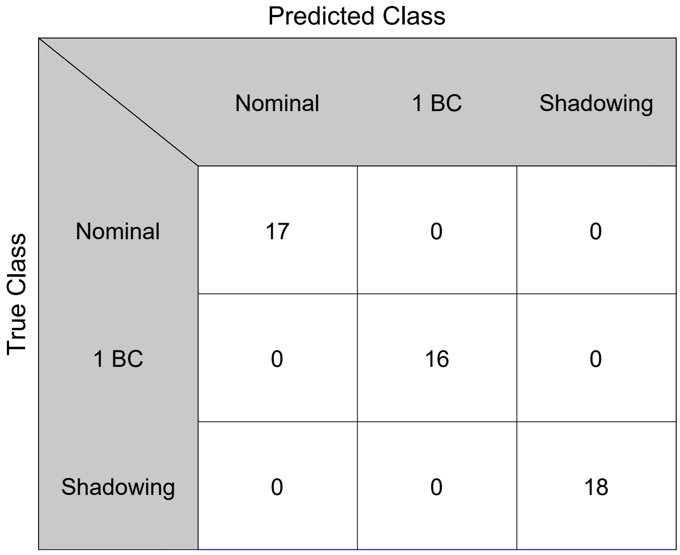


Figure 19: Training Data and Simulated Telemetry, Nominal

On the contrary, even if the three-dimensional instant value of the telemetry is sufficient for anomaly classification on the solar array power output, it cannot distinguish between permanently broken and temporarily shadowed cell. Indeed, this issue can only be addressed by training the SVM model over the entire cometary day, so that fluctuations in the power output are accounted for. To prove this consideration, a second SVM model is trained with multi-dimensional telemetry data that spans an entire cometary day. Three classes are considered: nominal, one broken cell and array periodic shadowing due to the presence of an obstacle. The simulation is run for several cometary days. Figure 20 shows the classification results relative to the testing subset. By considering a sufficient large time window, i.e. a cometary day, the model makes again correct predictions.



Predicted Class

True Class		Nominal	1 BC	Shadowing
	Nominal	17	0	0
	1 BC	0	16	0
	Shadowing	0	0	18

Figure 20: Confusion Matrix, case # 2

Part III

AUTONOMOUS ON-BOARD COLLISION AVOIDANCE

5 SPACE TRAFFIC MANAGEMENT

This chapter gives an introduction on the subject of Spacecraft Collision Avoidance, addressing needs and criticalities and explaining the relationship with Space Traffic Management. The proposed approach for collision warning is then described.

5.1 INTRODUCTION

Space is a huge but not an unlimited resource. It is getting more and more crowded because of the increasing number of space objects. While a desirable result from a scientific and engineering point of view, the consequence is on the other hand an increase in the probability of a collision, or communication failure, or some other catastrophic event. The phenomena was already brought to the attention of the scientific community since the 1970s, discussed as an agenda item of the UN Committee for the Peaceful Uses of Outer Space (COPUOS). Instead, the term *space traffic* or *space traffic management* as such, started appearing in relevant literature only since the turn of the century [13], and with the upcoming launches of mega-constellations (see for instance OneWeb) has seen very recently a new increase in popularity. Contributing to what discussed in Section 1.3, space traffic consists mainly of the motion and interaction of space debris, space vehicles and the use of radio frequency (RF) spectrum in outer space. While the focus is on improving international coordination and data sharing, it can be noticed from the available technical literature, such as Linares and Furfaro [25] and Peng and Bai [28], how great attention is given to Machine Learning technologies. Space debris consists of remnants from space vehicles, as well as natural objects like meteorites and planetary particles that travel through the Solar System. The exact number of man-made objects in space is not known but estimated to be in the tens of thousands: each of these objects, even if only a few millimeters in size have the potential of knocking out of operation a millions dollars-worth space vehicle. Figure 21 shows a snap-shot of the debris distribution around Earth. What can be expected in the foreseeable future are only more objects launched into space, because the right solution is not to put a stop on the progress of space engineering but to improve the available algorithms and methodologies. Mega-constellations have been mentioned already: they consists of several hundreds of satellites, which are likely to operate in the already populated LEO environment. As a reference, some of the characteristics of OneWeb constellation are

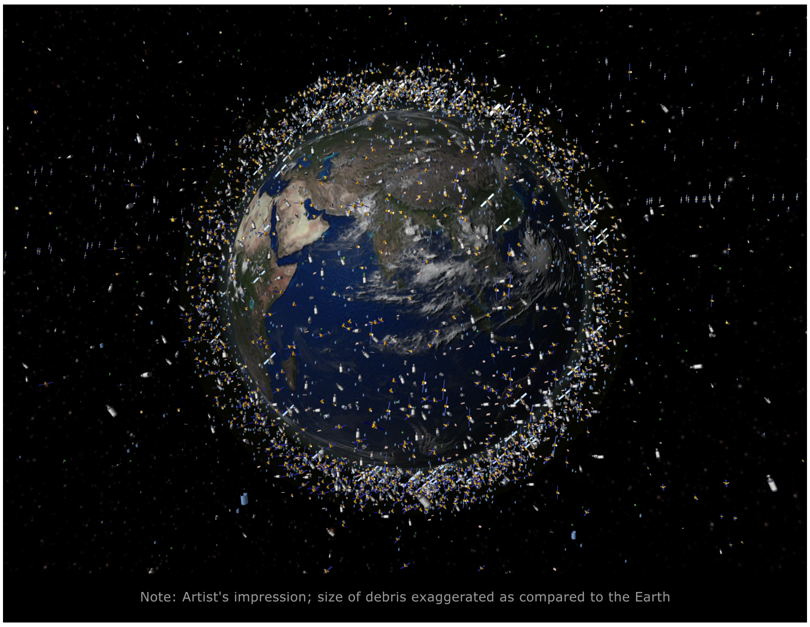

Note: Artist's impression; size of debris exaggerated as compared to the Earth

Figure 21: Debris objects in low-Earth orbit (LEO) ©ESA

given in table Table 5. The difficulty of collision avoidance planning and execution is going to step up a notch as a consequence, both in terms of raw complexity and human effort. Therefore, innovative methods and approaches in the area of autonomous decision-making, exploiting Machine Learning techniques, represent an interesting and valuable research topic.

Table 5: OneWeb design characteristics

Parameter	Units	Value
Satellite Mass	[Kg]	175-200
No. of orbital planes	[-]	18
Altitude	[km]	1200
Operational frequency	[GHz]	12-18

5.2 AUTONOMOUS COLLISION AVOIDANCE SYSTEM

As demonstrated by [2] [3], the subject is recently receiving more and more attention. Complexity of collision avoidance is significant as it must cover from the interaction with the entity(ies) providing space situational awareness (SSA) to the triggering and monitoring of collision avoidance maneuvers. In the process a number of crucial aspects need to be addressed: continuous screening and monitoring of conjunction warnings and assessment of their associated risk, close monitoring of high risk conjunctions in cooperation with the SSA entities, establishing criteria and decision making procedures for triggering collision avoidance maneuvers, designing appropriate maneuvers for each conjunction and verifying their safety with the SSA entities. Besides the complexity, the operational load associated to collision avoidance in LEO may also be significant. The large population of active satellites and space debris in this region may cause a substantial number of warnings, which render impractical and error-prone a human-based approach. The problem is further exacerbated if the operator is in charge of a large fleet of satellites. In this case automation is a must. The questions are then: How can these processes be automatized to a sufficient high degree? Is an on-board implementation feasible?

The problem of collision assessment between two spacecraft, from the determination of their expected position in the near future to the calculation of probability of collision, is a very complex one and difficult to be addressed in its entirety. The present work is then limited to the collision screening and warning part. A novel approach based on supervised learning is developed and then applied to a study case to test its performance.

5.3 COLLISION AVOIDANCE PROCEDURE

As a starting point, a brief overview of current collision avoidance procedure is hereby described, with a focus on the LEO environment. Satellite operators usually performs a monitoring several times a day in an automated process, detecting close approaches of operational LEO satellites against tens of thousands space objects listed in the Two-Line-Elements (TLE) catalog provided by USSTRATCOM. With reference to Aida, Kirschner, and Kiehling [4], the procedure usually consists of the following three steps:

1 First, the potential collision risk of the operational satellites is detected over 7 following days using a TLE catalog as well as precise orbit data of the operational satellites. Detected close approach events are listed in a report file, if the distance to a jeopardizing object is smaller than the predefined distance thresholds. Due to the uncertain nature of the measurements, the collision probability is also calculated for the potential close approach based on appropriate methods.

2 If the resulting collision probability exceeds the probability threshold (usually 10^{-4}), the collision risk is closely evaluated by analyzing the geometry at the time of the closest approach. In case a high collision risk is expected from the refined analysis, further orbit refinement through radar tracking could be foreseen as an additional step.

3 Finally, the close approach event is further analyzed based on the precise and latest orbit information, and a collision avoidance maneuver is planned if required. Usually the maneuver takes place ca. half an orbit before the expected close approach, then another one ca. half an orbit later is executed to correctly reposition the spacecraft in the appropriate orbital slot.

As it can be deduced from the description above, the procedure is laborious and computationally intensive. In an effort to reduce the computational demand, moving towards an autonomous implementation, one can think to carefully import and compare recent strategies in aerial collision avoidance. As anticipated in Section 1.3, Kochenderfer and Chryssanthacopoulos [24] used dynamic programming for creating the ACAS Xu horizontal decision making logic. It results in a large numeric lookup table that contains scores associated with different maneuvers from millions of different discrete state. Julian, Lopez, et al. [23] explores instead two approaches for compressing the score table (thus permitting an on-board implementation) without loss of performance as measured by a set of metrics: origami compression and, more interestingly, a robust non-linear function approximation that represents the table using a deep neural network, schematized in Figure 22.

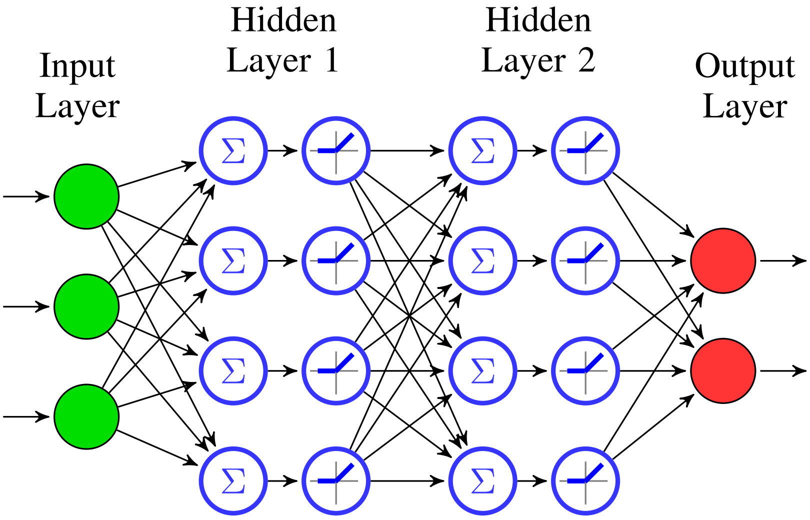


Figure 22: Neural Network Diagram [23]

Returning to spacecraft collision warning, an effort is made to preliminary explore the feasibility of an approach of this kind, i.e. to build a Support Vector Machine model good enough to capture the complex and non-linear dynamics that governs the motion of a spacecraft. Ideally, the procedure would be something like Figure 23, which simplistically summarize what discussed in the previous paragraphs. Clearly, as a preliminary analysis, the problem has to be greatly simplified. The idea of building a one-for-all model, that addresses all

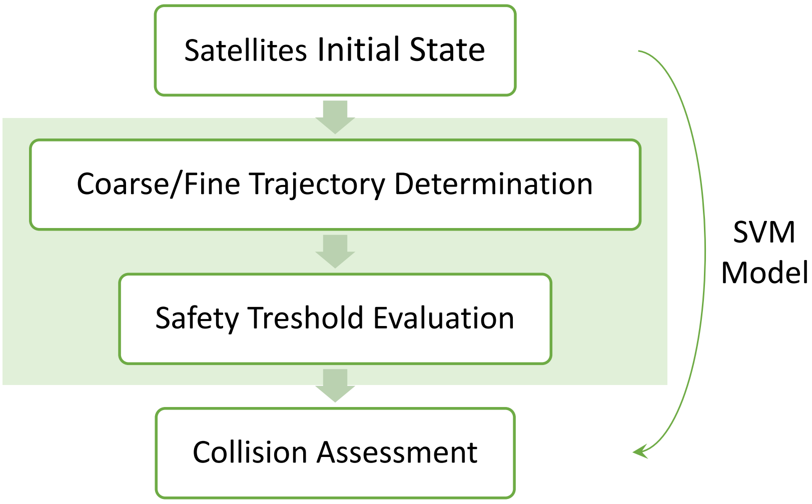

Satellites Initial State
Coarse/Fine Trajectory Determination
Safety Treshold Evaluation
Collision Assessment
SVM Model

Figure 23: Basic idea of the procedure

possible collision geometries, seems unfeasible. On the contrary, application specific models could theoretically produce satisfactory results while preserving the necessary generalization properties. Other assumptions are instead made to simplify the speed of execution of the numerous trials, required in the conceptual phase of the development. They are listed in the following:

- Circular, High-Inclination, Low Earth Orbits: driven by the choice of the application case, a satellite mega-constellation (see Chapter 6).
- Short-term collision warning, i.e. less than one orbit before the predicted close approach.
- Relative distance threshold as the metric for collision assessment.
- Non-perturbed 2-Body dynamics: undoubtedly the strongest assumptions of them all, justified again by the fact that the present work represents the first stepping stone towards a more realistic implementation.

The following section describes how the training data, fed to the SVM classifier, are built.

5.4 DESCRIPTION OF THE MODEL

5.4.1 *Collision Geometries*

The state of an object in a 3D space is defined by six independent variables. A common choice for a satellite is represented by the Classical Orbital Elements vector $[a, e, i, \Omega, \omega, \theta]$, where in the case of a Earth-orbiting satellite

a Semi-major axis, defines the size of the orbit.

e Eccentricity, Defines the shape of the orbit.

i Inclination, defines the orientation of the orbit with respect to the Earth's equator.

Ω Right ascension (RA) of the ascending node, defines the location of the ascending and descending orbit locations with respect to the Earth's equatorial plane.

ω Argument of perigee, defines where the low point, perigee, of the orbit is with respect to the Earth's surface.

θ True anomaly, defines where the satellite is within the orbit with respect to perigee.

In order for the SVM model to discern between close/safe approaches, without the need of propagating forward the satellite states, a proper training set must be created. Assuming all six of them as free variables would result in an intractably large matrix. Instead, under the assumptions discussed in the previous section, the following considerations can be made use of:

- The geometrical shape of the orbit is fixed in time, therefore a, e are determined by the problem under consideration.

- In the case of circular orbits it can be assumed that the periapsis is placed at the ascending node and therefore $\omega = 0$.

- The close approach between the main and secondary spacecraft is going to occur in the neighborhood of one of the two nodes along the line of intersection of the two orbital planes. The nodal axis is identified by Equation 17:

  $$\mathbf{n} = \mathbf{h}_1 \times \mathbf{h}_2 \tag{17}$$

  where $\mathbf{h} = \mathbf{r} \times \mathbf{v}$ is the specific angular momentum vector and $\mathbf{r}$, $\mathbf{v}$ are the position vector of the satellite relative to the center of the earth and its derivative, respectively. It is straightforward to compute the angle between the node line of each orbit and $\mathbf{n}$ (called β for future reference). The true anomaly is then set equal to this value and an additional parameter δ is introduced, which represents a slight displacement of the secondary from the aforementioned reference position.

The remaining two parameters, i and Ω, are free variables. Anticipating that the application example revolves around collision warning for a mega-constellation, the range of the three free variables $[i, \Omega, \delta]$, together with the value of the other fixed parameters, is chosen to be

Table 6: Parameters range

Parameter	Units	Value/Range	No. of samples
a	[km]	7571	
e	[-]	0	
i	[deg]	[80, 89], lin. spaced	15
Ω	[deg]	[0, 180], lin. spaced	15
ω	[deg]	0	
δ	[deg]	[-1, 1], exp. spaced[1]	81

compliant with Table 5. Using the values in Table 6, all combinations are exhausted and a matrix as the following one is created:

$$\begin{bmatrix} [a, e, i, \Omega, \omega, \theta]_{p,1} & [a, e, i, \Omega, \omega, \theta]_{s,1} \\ [a, e, i, \Omega, \omega, \theta]_{p,2} & [a, e, i, \Omega, \omega, \theta]_{s,2} \\ \dots & \dots \end{bmatrix}$$

5.4.2 *Propagation Routine*

First $\mathbf{r}, \mathbf{v}$ are derived from the COE vector. Then, the well-known and simple method of Lagrange coefficients is employed to determine the position and velocity of a spacecraft from the initial conditions. The equations are summarized in the following [14]:

$$\begin{aligned} \mathbf{r} &= f\mathbf{r}_0 + g\mathbf{v}_0 \\ \mathbf{r} &= \dot{f}\mathbf{r}_0 + \dot{g}\mathbf{v}_0 \end{aligned} \tag{18}$$

with

$$\begin{aligned} f &= 1 - \frac{\mu r}{h^2}(1 - \cos\Delta\theta) \\ g &= \frac{r r_0}{h}\sin\Delta\theta \\ \dot{f} &= \frac{\mu}{h}\frac{1 - \cos\Delta\theta}{\sin\Delta\theta}\left[\frac{\mu}{h^2}(1 - \cos\Delta\theta) - \frac{1}{r_0}\frac{1}{r}\right] \\ \dot{g} &= 1 - \frac{\mu r_0}{h^2}(1 - \cos\Delta\theta) \end{aligned} \tag{19}$$

and

$$r = \frac{h^2}{\mu}\frac{1}{1 + \left(\frac{h^2}{\mu r_0} - 1\right)\cos\Delta\theta - \frac{h v_{r0}}{\mu}\sin\Delta\theta} \tag{20}$$

[1] This choice is made so that a finer grid around close approaches situations is created

5.4.3 *Feature Choice and Pre-Processing*

After propagating each scenario in Section 5.4.1 and labeling it as belonging to class 1 for close approaches and class 0 for the rest (the minimum relative distance is set to $D = 25\text{km}$), the raw matrix of predictors is created. Based on empirical tests, a good choice for the attributes is given by:

$$X_{\text{pred},j} = \left[\, [a, e, i, \Omega, \omega]_{p,j}, [a, e, i, \Omega, \omega]_{s,j}, \delta_j \, \right] \tag{21}$$

The angle δ is easily obtainable as a function of the spacecraft true anomalies, inclinations and right ascensions of the ascending node. The predictors are standardized using their corresponding means and standard deviations and then split into Training Set (75%) and Validation Set (25%). Finally, the Training Set is balanced using the procedure in Section 2.4.

6 ANALYSIS OF THE RESULTS - PART II

This chapter summarizes the performance of the model previously described, showing both training and validation accuracy. In addition, a satellite mega-constellation is simulated and the model is tested against an injected failure.

6.1 TRAINING AND VALIDATION ACCURACY

Let us define the processed training and validation set of Chapter 5 as $[X_{tr}, Y_{tr}]$ and $[X_{val}, Y_{val}]$, respectively. The first one is fed to a Support Vector Machine classifier, while the second one is used at the end of the procedure to evaluate its performance. Again, the metric is the number of correctly classified instances. The outcome of the training is then reported in Table 7.

Table 7: Summary of results

Parameter	Value
Kernel function	RBF
C	256
γ	0.83
Cross-validation	10-fold
Accuracy	Training: 99.8% Validation: 99.8667%

6.2 APPLICATION TO SATELLITE MEGA-CONSTELLATION

It is interesting to analyze how the model fares against an artificial failure injected in a satellite mega-constellation. As a back-up safety tool, a support vector network with the ability to instantaneously issue a warning, in the eventuality of a close approach between two spacecraft of the constellation, would be desirable. Indeed, as the number of satellites grows, the optimal phasing requirement tighten up. Even a small displacement from the nominal position could be problematic. The system shares the same characteristics as the ones reported in Table 5.

The satellite characterized by $\Omega = 0$ and $\theta = 0$ is taken as a reference. Then, after a quick analysis, the following arbitrary slight variation is introduced to the optimal phasing between spacecraft.

$$phasing = 0.46\,(2\pi/noSatPerPlane/noPlanes) \tag{22}$$

The minimum distance, over the span of one orbit, that the reference satellite experiences against one of the others vary according to Figure 24. For all the satellites that shares a close passage with the reference spacecraft, the relative predictors are extracted and fed to the trained model. In this case also, the warning is issued correctly.

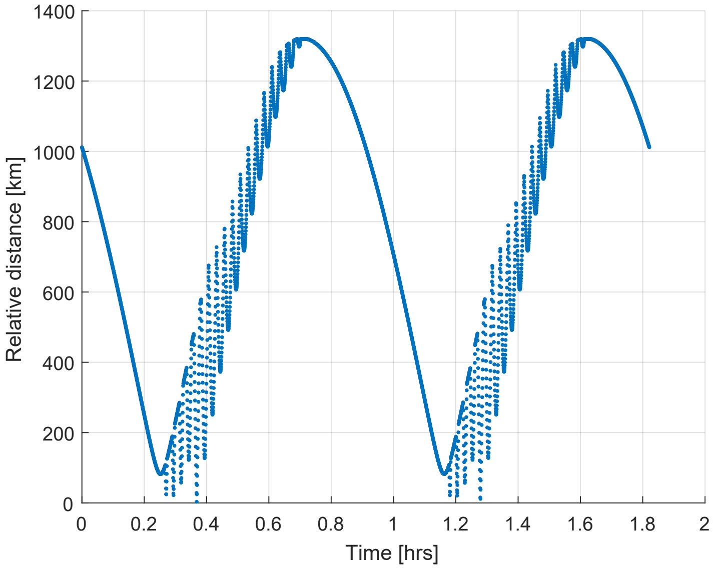


Figure 24: Minimum relative distance between reference satellite and the rest of the constellation over time

Part IV

FINAL REMARKS

7 CONCLUSIONS AND PROSPECTIVE WORK

At the end of the present thesis exposition, it is worth to discuss the methodologies implemented and the results presented. Moreover, it is significant to suggest prospective work related to the study proposed.

7.1 CONCLUSIONS

The scope of the thesis was to assess the feasibility of Multi-Class Support Vector Machine for anomaly classification, with application to Rosetta's lander Philae Solar Generator. A complete model of the lander solar generator was developed, in order to simulate the produced telemetry accurately enough for the purpose of fault classification. The analysis shows that the available measurements (only two) are not enough. Instead, adding an artificial third attribute and according to the model and assumptions described, autonomous fault detection via a machine learning approach can be a successful option. The procedure is rather general, though, and the model described in Chapter 3 can be readily applied to an arbitrary solar generator system.

In addition, the topic of Space Traffic Management has been explored, addressing the need of automatization in the space sector. A novel approach to preliminary collision warning was proposed, based on supervised learning, with the intent of bringing closer together the artificial intelligence and space engineering domains. The initial results appear to be positive.

7.2 PROSPECTIVE WORK

There are also directions for future works. The anomalous detection study can be further improved by implementing a more complete and accurate model of Philae power generation subsystem, so to take into account a greater number of faults. More so, this approach requires the complete knowledge of the nominal and fault cases: additional improvements can be made, in this regard, by studying the applicability of one-class classification for nominal/anomalous behavior detection as an additional first step in the procedure. For what regards autonomous collision warning, the first step would be to assess the performance of the approach under more realistic conditions. Then, if the results are encouraging, one can think of adding a Support Vector Regression model to estimate magnitude and direction of a sub-optimal collision avoidance maneuver.

BIBLIOGRAPHY

[1] URL: https://towardsdatascience.com/cross-validation-in-machine-learning-72924a69872f.

[2] URL: https://techport.nasa.gov/datasheet/93252.

[3] URL: https://artes.esa.int/funding/autonomous-collision-avoidance-system-ngso-artes-3a093.

[4] S. Aida, M. Kirschner, and R. Kiehling. «Collision Avoidance Operations for LEO Satellites Controlled by GSOC.» In: *SpaceOps 2010 Conference* (2010), pp. 1–10.

[5] S. Armstrong and W.G. Hurley. «A thermal model for photovoltaic panels under varying atmospheric conditions.» In: *Applied Thermal Engineering* 30.11 (2010), pp. 1488–1495.

[6] H. Brennan and American Institute of Aeronautics and Astronautics. *International Space Cooperation: Addressing Challenges of the New Millennium, Report of an AIAA, UN/OOSA, CEAS, IAA Workshop, March 2001*. American Institute of Aeronautics and Astronautics, Mar. 2001.

[7] O Brown, T. Cotton, et al. *Orbital Traffic Management Study*. Tech. rep. SAIC, Nov. 2016.

[8] G. M. Caputo. «On-comet attitude determination of Rosetta lander Philae through nonlinear optimal system identification.» MA thesis. Politecnico di Milano, 2012.

[9] M Cattafesta. «Modelling and simulation of Rosetta Lander Philae solar arrays.» MA thesis. Politecnico di Milano, 2013.

[10] C. Chang and C. Lin. «LIBSVM: A library for support vector machines.» In: *ACM Transactions on Intelligent Systems and Technology* 2 (3 2011). Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm, 27:1–27:27.

[11] C. Contant-Jorgenson, P. Lála, K. Schrogl, et al. *Cosmic Study on Space Traffic Management*. Tech. rep. International Academy of Astronautics (IAA), 2006.

[12] C. Cortes and V. Vapnik. «Support Vector Networks.» In: *Machine Learning* 20.3 (1995), pp. 273–297.

[13] H. Cukurtepe and I Akgun. «Towards space traffic management system.» In: *Acta Astronautica* 65 (Sept. 2009), pp. 870–878.

[14] H. D. Curtis. *Orbital Mechanics for Engineering Students*. 2005.

[15] C. R. Frost. «Challenges and Opportunities for Autonomous Systems in Space.» In: *National Academy of Engineering's U.S. Frontiers of Engineering Symposium*. New York, Sept. 2010.

[16] Y. Gao et al. «Fault detection and diagnosis for spacecraft using principal component analysis and support vector machines.» In: 7th IEEE Conference on Industrial Electronics and Applications (ICIEA). 2012, pp. 1984–1988.

[17] F. Gaspari. «Numerical and experimental characterization of Philae's photovoltaic balcony panel.» MA thesis. Politecnico di Milano, 2016.

[18] Haibo H. et al. «ADASYN: Adaptive synthetic sampling approach for imbalanced learning.» In: *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. June 2008, pp. 1322–1328.

[19] Chih-Wei Hsu and Chih-Jen Lin. «A comparison of methods for multiclass support vector machines.» In: *IEEE Transactions on Neural Networks* 13.2 (2002), pp. 415–425.

[20] M. Jah. *Disruptive Research Ideas for Space Situational Awareness and Space Traffic Management Needs*. Sept. 2016.

[21] A.D. Jones and C.P. Underwood. «A thermal model for photovoltaic systems.» In: *Solar Energy* 70.4 (2001), pp. 349–359.

[22] A. Jónsoon et al. «Autonomy in Space: Current Capabilities and Future Challenges.» In: *AI Magazine (© AAAI)* 28.4 (2007), pp. 27–42.

[23] K. Julian, J. Lopez, et al. «Policy compression for aircraft collision avoidance systems.» In: IEEE/AIAA 35th Digital Avionics Systems Conference (DASC). Sept. 2016, pp. 1–10.

[24] M. J. Kochenderfer and J. P. Chryssanthacopoulos. *Robust Airborne Collision Avoidance through Dynamic Programming*. Tech. rep. Lincoln Laboratory, Massachusetts Institute of Technology, 2011.

[25] R. Linares and R. Furfaro. «Space Object classification using deep Convolutional Neural Networks.» In: 19th International Conference on Information Fusion. 2016.

[26] G. Mengali and A. A. Quarta. *Fondamenti di Meccanica del Volo Spaziale*. Pisa University Press, 2013.

[27] K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[28] H. Peng and X. Bai. «Improving Orbit Prediction Accuracy through Supervised Machine Learning.» In: (2018), pp. 1–30.

[29] G. Pinzan. «Landing Site Selection for Rosetta Lander Philae through a Multidisciplinary Approach.» MA thesis. Politecnico di Milano, 2012.

[30] F. Topputo, F. Bernelli-Zazzera, and A. Ercoli-Finzi. «Power Production for Small Body Landers: Post-Launch Activities on Philae's Power Subsystem.» In: 62nd International Astronautical Congress. Cape Town, SA, 2011.

[31] Y. Zhao et al. «Decision tree-based fault detection and classification in solar photovoltaic arrays.» In: 27th Annual IEEE Applied Power Electronics Conference and Exposition (APEC). 2012, pp. 93–99.